Secure Network Bridge
Project Proposal

Kevin Arunski
Jonathan Fears
John Rose
Billy Runyan

CPSC 483
Fall 2000

TABLE OF CONTENTS

Abstract 3
Project Background 4
Project Objectives and Deliverables 6
Block Schematic 7
Hardware System Architecture 8
Software System Architecture 10
Component List – Prototype 12
Component List – Test 12
Week to Week Goals 13
Team Member Responsibilities 14
Special Test Environments 15
Hardware / Software Percentage 16
Cost 17
References 18

Abstract

We plan to develop a device, that when implemented and operated correctly, will transmit data between two networks in a single direction, called the SNB. The SNB will allow a user operating on a high-security network, such as the one within the CIA, to download information from a low-security network, namely the World Wide Web or other LAN. The device will ultimately provide absolute assurance that no information can be transmitted in the reverse direction, from the high-security network to the low security-network.

The SNB will be composed of roughly 65% software and 35% hardware. The hardware for the SNB is based around a 486 CPU. The software portion of the device is simple in theory but will ultimately determine the ease of use and overall functionality of the SNB. Applications will be necessary for both sending and receiving files, and administering settings on the unit.

The primary goal we wish to accomplish by developing the Secure Network Bridge, in addition to fulfilling the functionality requirements, is to create an effective product that occupies a minimum amount of space, is simple to implement, and inexpensive relative to similar products.

Project Background

Emerging computing technologies have revolutionized the way we store and transfer information. The connectivity of the Internet brings with it an invasive element, the possibility of a user across the world compromising your privacy. Such concerns become grave when one considers the implications of insecure national security information.

New technologies require new security measures to ensure the necessary amount of isolation from the outside world. Business users frequently need information from the Internet, but cannot allow information to be taken from their system. Current implementations involve the use of isolated LAN networks for secure data, with no connection to the outside world. We seek to satisfy the need for information transfer from outside of the LAN while ensuring that no secure data leaves the LAN.

This project will service a need for many high-security computing environments. We define high-security as being completely inaccessible from outside the physical location of the computing environment. In this case, the computing environment is a LAN, composed of workstations and servers connected via an Ethernet or token-ring network.

Organizations that use these sorts of high-security networks frequently need to access information on an open, unsecured network such as the Internet. Another example would be a separate, low-security network connected to the Internet through a firewall. Information gathered on these low-security networks is considered sensitive; however, it must be carried out on the high-security network. Thus, there exists a need to transfer information from the low-security network to the high security network without compromising the security level of the high security network.

An example of this situation is the Wen-Ho Lee case that has been in the news recently. Lee, was a scientist at the Los Alamos National Laboratory. Lee was indicted recently on the charge of espionage. According to officials, he downloaded information from the Laboratory's high security network onto his personal office computer, where he thought it would be easier to work on. The Secure Network Bridge aims to prevent this situation. If there were only a SNB connecting the two networks, Lee would not have been able to transfer data from the low security network to the high security network.

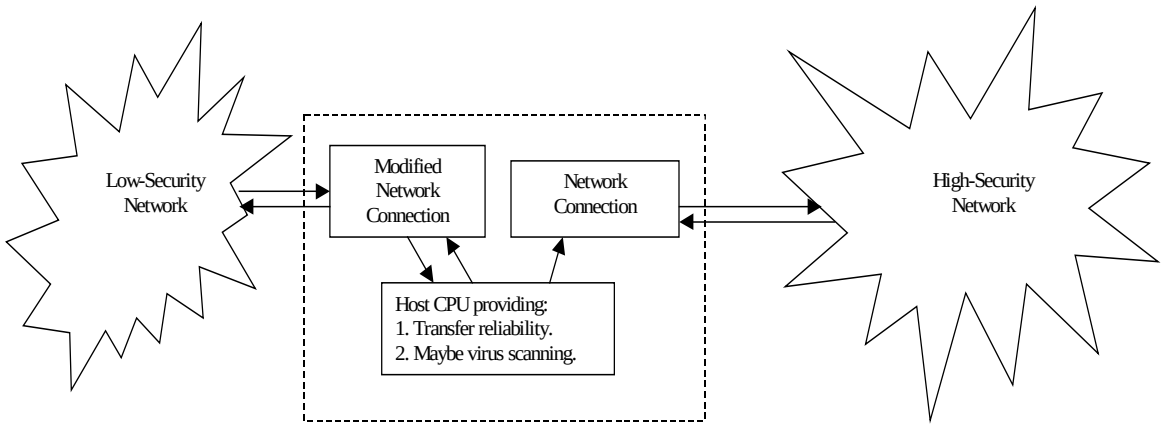
Project Objectives and Deliverables

The objective of this project is to provide a product that allows for a secure and reliable one-way data path for communications between two networks.

There will be several deliverables resulting from this project:

1. First, there will be a single box containing the modified network connection for the low security network, the normal network connection for the high security network, and the host CPU. This will be the “Secure Network Bridge”.
2. Also client software will be created for both the low and high security networks. This will provide a Web-based or graphical interface to make using the Secure Network Bridge user-friendly.
3. A user manual will accompany the product. This user manual will describe in detail the operation of the SNB, including how to administer it and how to use the client software.

Block Schematic



A low-security network user issues a request to transfer data. The client software sends data from the low-security network to a network connection in the Secure Network Bridge. The SNB authenticates the client connection based on an internal username and password list. The client software on the low security network then creates a package containing the file or files to be sent, and sends it to the Secure Network Bridge. The Secure Network Bridge then initiates a connection to the client software on the high-security network. The high-security client unpacks the files, and checks their integrity. If the files are verified, the application makes them available to the user.

Hardware System Architecture

Our requirements describe an inexpensive, embedded system. One can accomplish all the functions that it performs with an ordinary PC. However, to control the configuration and operation of the device, the SNB hardware will have a limited set of capabilities. Also, the specific nature of the SNB functionality will reduce the necessary cost and size of the device so that it is smaller than an average PC.

The SNB will rely on a 486 CPU. We choose this CPU because with price and availability issues in mind. Also, the 486 will provide application compatibility. Should the user of the SNB require additional features, such as virus scanning, they can be added in using existing software applications. The 486 is also easily replaced with a faster Pentium or later CPU, if features such as encryption require it.

The SNB's operating system and application software will reside on an 8MB Compact Flash memory device. We chose this device because it will be easy to implement. The Compact Flash device appears to the operating system as an IDE drive. Therefore, we can develop our prototype using a small partition on an IDE drive. The SNB's operating system and application software will require a total of 8 MBs of RAM.

Two Ethernet controllers will serve as network connections. These controllers will interface with the 486 via an ISA bus interface, at least in the prototype SNB. The SNB will contain a serial port, as well. This port will provide a console for administering settings within the SNB. For example, the SNB will need network settings for both of its interfaces. Therefore some local interface to the device must exist. The serial console has several advantages. First, it is inexpensive to implement. Second, most laptop and desktop computers contain a serial port that can communicate with the SNB. Finally, by keeping the administration duties

separate from the operation of the device, we increase the security of the system. For example, once the SNB is disconnected, one can disconnect the console from the serial port without any necessary changes to the system.

We hope to implement the serial and network controllers in an FPGA device, thus reducing the number of chips in the system. This will reduce the size, cost, and complexity of the SNB.

Software System Architecture

The SNB itself needs an operating system to manage communications from the network interfaces, and to run the application software that will provide an authentication mechanism for transfers. We are going to use Linux, for several reasons:

1. It is inexpensive.
2. It is open-source. This is very important, since it allows us to easily make changes to the operation of the network interfaces. For example, we may want to modify them so that no data can be sent from the high side to the low side.
3. It is small. A pared-down operating system should be able to fit within the 8MB of flash we have allocated. Also, the Linux kernel uses less than 4MB of RAM, leaving adequate room for our application software.

On the SNB, there will be two applications. This first application will handle the transfer of files through the system. It will provide authentication for the low security client software. Then it will receive data from the low security client, and initiate a connection to the high-security client. Finally, the transfer manager sends the data it receives from the low security client to the high security client.

An administration program also runs on the SNB. This program will provide a menu-driven interface to change settings within the SNB. This program will be accessed through a serial port. This console can administer two types of settings:

1. Network configuration. Both network interfaces will be configured through this program. The IP addresses, for example will be set here.
2. User administration. This feature will add users to the SNB, and set their passwords.

Both the low security and high security networks will run client software for accessing the SNB. We plan to implement this software in Java, so that it can run on virtually any platform, or be embedded into a web page. On the low

security side, the client will provide the user with the ability to chose which file or files they wish to send. Furthermore, it will ask the user which computer on the high-security network they would like to send the files to. Before sending the files, it will also prompt the user for their username and password, which it will forward to the SNB for authentication.

On the high security side, the client software will display what files it has received from the SNB, and allow the user to chose where to store them on their computer's file system.

Component List – Prototype

- 486 CPU and related chipset
    - A FPGA will later be used to provide chipset.
- Ethernet Controller x 2
    - A FPGA will later be used to provide Ethernet
- Serial Communication Controller
    - To connect a serial console
- 8MB DRAM, 8MB IDE Drive Space
    - Drive will be replaced with Compact Flash

For our prototype, will be using a 486 PC to simulate the hardware that will be used in the actual SNB. We will also use a more modern PC as a development system.

Component List – Test

- Prototype SNB
- One networked computer to simulate the low-side client
- One networked computer to simulate the high side client and to connect to the Administration Console
- Two Ethernet cross-over cables, or two Ethernet networks
- A null-modem cable to connect to the Administration Console

We will need one more PC in addition to our development and prototype systems. This one will simulate the high security network client. We will use one of the networked PC's to connect to the serial port and the administration console, as well as act as a client computer.

Week to Week Goals

Week of …	Goals
9/19	- Finish proposal
9/27	- Obtain the SNB prototype box - Set up a test system for development use
10/3	- Bi-weekly Report 1 - Finalize Protocol for client software
10/10	- Begin work on client software - SNB should be able to move data from one network to the other
10/17	- Bi-weekly Report 2 - Begin client-SNB protocol implementation
10/24	- Complete mid-term presentation
10/31	- Continue software implementation - Begin testing
11/7	- Bi-weekly Report 3 - Begin Administration Console
11/14	- Continue Administration Console - Begin migration to FPGA - Work on User Manual
11/21	- Bi-weekly Report 4 - Finish \ test Administration Console - Continue FPGA migration - Finish on User Manual
11/28	- Thanksgiving Holiday - Work on final presentation
12/07	- Final presentation / demo

Team Member Responsibilities

Member	Responsibilities
Kevin Arunski	- Put together SNB hardware prototype - Write SNB transfer manager - Write User Manual
Jonathan Fears	- Write SNB client software - High side client. - Write Administration Console
Billy Runyan	- Write SNB client software - Authentication system - Write Administration Console
John Rose	- Write SNB client software - Low side client - Work on FPGA component integration - Write User Manual

Special Test Environments

- 1 SNB box
- 2 Client computers, one representing each network
- 2 cross-over Ethernet cables, or two Ethernet networks
- 1 null modem cable.

Our test environment requires two systems to fully test the functionality of the system. During development, we will use a single PC to simulate the operation of the low side and the high side systems.

Hardware / Software Percentage

Approximately 65% Software

- Low Security Client
    - Written in Java
    - Makes connection to SNB
- SNB Operating Environment
    - User-level program to manage transfers
    - Administration program for Serial Console
- High Security Client
    - Written in Java
    - Receives connections from SNB

Approximately 35% Hardware

- Interfacing FPGA with 486 CPU + Chipset

By using components that already exist, we will have a design that is more open to future expansion and customization.

Cost

- Prototype based on 486 CPU
  - $25 for a used PC
  - $30 for ISA network cards

Our initial prototype can be implemented using standard PC hardware.

- Prototype using FPGA
  - $250 for ISA FPGA development board
  - Intellectual Property Costs.

The FPGA prototype will replace the network controllers and serial ports with a single ISA board containing an FPGA.

References

- Wen Ho Lee Indicted
  - http://www.abcnews.go.com/sections/world/DailyNews/wenholee991211.html
- APS FPGA Price List
  - http://www.associatedpro.com/apsprice.html
- Hitachi Flash Memory Systems
  - http://www.halsp.hitachi.com/flashcards/